


CS2100

Doubly linked
lists



Recap

- HW & lab due over the weekend

Aside: HW suggestion

refcount is useful for debugging

in dump:

```
cout << walk -> element << " " <<
walk -> ref-count
<< " " ;
```

- Review Monday
- Test Wednesday

Lists: Motivation

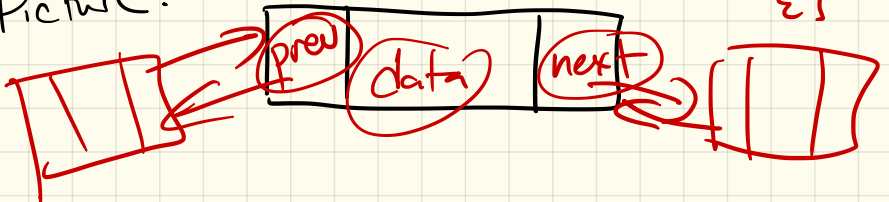
Insert in vectors is slow!
If I'm changing 1 thing,
want O(N).

Doubly linked List struct:
template <typename T>

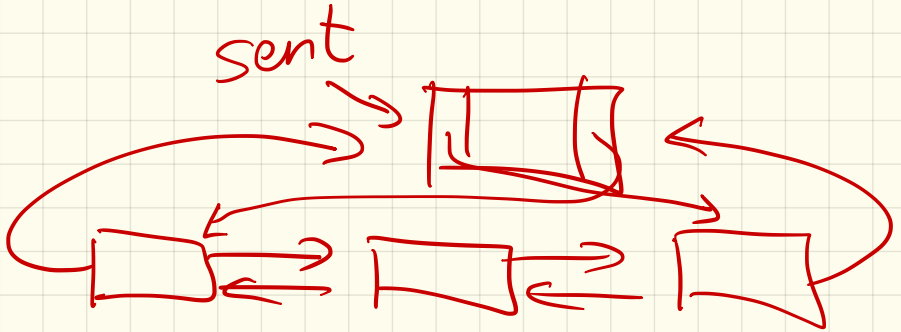
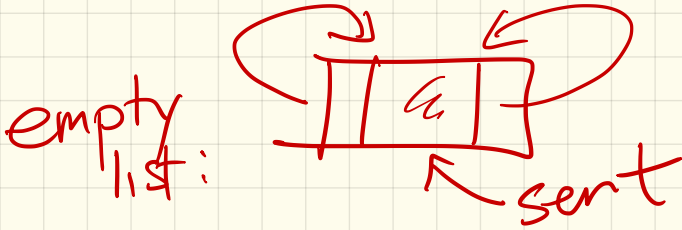
```
struct Node {  
    T _data;  
    Node* _next;  
    Node* _prev;
```

```
} Node(T& d = T(), Node* p = NULL,  
      Node* n = NULL) :  
    _data(d), _next(n), _prev(p)  
    {}
```

Picture:



Circularly linked lists:
(w/ sentinel node)

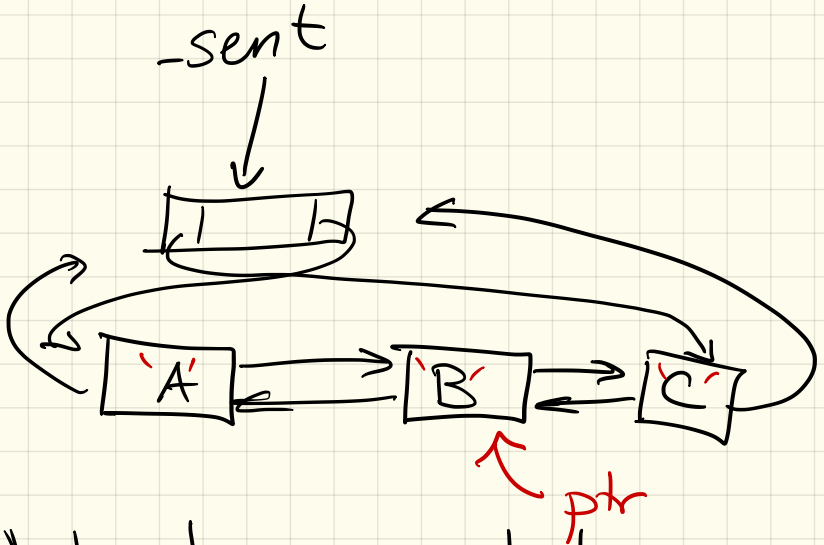


private variables:
(of List)

```
Node* _sent;  
unsigned int size;
```

Issue: Pointers

Think about insert:



What do we need to send as input?

location + data

In C++, pointers should be kept hidden & used cautiously.

Why?

-security

So: we need a wrapped up pointer, set up to avoid seg faults.

(Think vector vs. array.)

We'll call these iterators.

(see code)

```
class Iterator {  
    private:  
        Node* _current;  
  
    public:  
        // constructor  
  
        operator ++  
  
        operator --  
  
}
```